Newnes

移动与嵌入式开发技术

# Hardware/Firmware Interface Design:
## Best Practices for Improving Embedded Systems Development

# 硬件/固件接口设计

## ——提高嵌入式系统开发效率的最佳实践

- 减少ASIC、ASSP、SoC及FPGA等产品的开发延时
- 讲授、演示原则和最佳实践
- 为常见问题和缺陷提供解答以节省时间和精力

(美) Gary Stringham 著
张 鼎 贺小川 译

ELSEVIER
清华大学出版社

移动与嵌入式开发技术

# 硬件/固件接口设计

## ——提高嵌入式系统开发效率的最佳实践

(美) Gary Stringham 著

张 鼎 贺小川 译

清华大学出版社

北 京

Hardware/Firmware Interface Design：Best Practices for Improving Embedded Systems Development
EISBN：978-1-85617-605-7


ISBN：9789812727114


**图书在版编目(CIP)数据**

硬件/固件接口设计——提高嵌入式系统开发效率的最佳实践/(美)斯特林汉姆(Stringham, G.)著；张鼎，贺小川 译. —北京：清华大学出版社，2011.11
(移动与嵌入式开发技术)
书名原文：Hardware/Firmware Interface Design：Best Practices for Improving Embedded Systems
ISBN 978-7-302-26701-0
I. 硬… II. ①斯… ②张… ③贺… III. 微处理器—系统设计　IV. TP332

中国版本图书馆 CIP 数据核字(2011)第 181090 号

**责任编辑**：王　军　谢晓芳
**装帧设计**：牛艳敏
**责任校对**：成凤进
**责任印制**：何　芊
**出版发行**：清华大学出版社　　**地　　址**：北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**邮　　编**：100084
**社　总　机**：010-62770175　　**邮　　购**：010-62786544
**投稿与读者服务**：010-62776969，c-service@tup.tsinghua.edu.cn
**质　量　反　馈**：010-62772015，zhiliang@tup.tsinghua.edu.cn
**印 装 者**：清华大学印刷厂
**经　　销**：全国新华书店
**开　　本**：185×260　**印　张**：18　**字　数**：438 千字
**版　　次**：2011 年 11 月第 1 版　　**印　　次**：2011 年 11 月第1 次印刷
**印　　数**：1～4000
**定　　价**：39.00 元

产品编号：037454-01

# 译 者 序

如今，嵌入式产品已经渗入日常生活和工业生产的各个领域。数码产品、家用电器、工业设备、交通及通信工具中都随处可见嵌入式产品的身影。嵌入式产品与传统电气产品的关键区别就在于包含了可编程逻辑器件，因此其开发通常包含硬件电路设计以及针对可编程逻辑器件的软件设计。这种软件设计与传统的软件设计类似，但是与硬件的相关性更大，通常称为固件设计。嵌入式产品需要固件与硬件的配合才能工作，因此，二者间的接口设计至关重要。

尽管硬件与固件接口的重要性已经是业内公认的关键技术，但是有关这方面的技术资料大都是针对某个系统的详细设计，或者是个人经验，尚缺乏系统的讨论，特别是缺乏通用性的设计方法。本书在这方面做出了开创性工作，提出了硬件和固件设计的七项原则，并从工程实际中提炼出 300 多个最佳实践。由于本书作者的技术背景和宝贵的工作经历，使得这些原则和实践的可用性极高，是难得的技术财富。相信本书必能使广大硬件和固件设计人员获益匪浅。

本书主要由张鼎翻译，贺小川翻译了部分内容。译文虽然经过多次修改和校正，但是由于译者水平有限，加之时间仓促，疏漏和错误在所难免，真诚地希望同行和读者不吝赐教，不胜感激。

# 前　言

您既可以找到由硬件工程师编写的向硬件工程师们讲授如何设计硬件的书籍，也可以找到由固件工程师编写的向固件工程师们讲授如何设计固件的书籍。本书由固件工程师编写，但是主要面向硬件工程师。

很多工程师在硬件上运行固件时遇到过问题。硬件和固件通常是分别独立设计而成并期望在集成时能够一起工作。但是，这会出现一些问题和缺陷。有时甚至不知道缺陷在何处——是硬件还是固件的问题，或者是文档的问题。

有关如何使硬件和固件在一起协同工作的书籍非常少。本书试图填补这一空白。本书阐述了硬件领域和固件领域之间的接口，并提供实践以减少生产嵌入式系统产品所需的时间和工作量。它涵盖了开发硬件/固件接口相关的所有方面，包括开发的过程、顶层设计以及详细设计等。

本书的主要特色是包含 300 多个最佳实践，可为开发过程及设计的各个方面提供详细的指导。这些最佳实践的应用效果极佳，但是它们只适用于特定的情况。应当根据实际情况详细地对照检查它们。全书处处强调让工程师形成他们自己的最佳实践集。尽管工程师是以这 300 多个最佳实践作为起点的，但是应当以此为基础不断发展，逐渐形成他们自己的最佳实践集，因为这样做可以增加其所在组织的项目获得成功的几率。

为了帮助工程师理解这 300 多个最佳实践，同时为了帮助他们创建自己的最佳实践集，本书列出了七项原则作为纲领性指导。即使在某些情况下没有特别合适的最佳实践，这些原则也能帮助工程师遵循正确的方向工作。依靠这七项原则和 300 多个最佳实践，将提高设计小组成功生产嵌入式系统产品的能力。

## 本书概述

下面的各章概述提供本书内容的概述，可以帮助读者浏览全书。

第 1 章：该章为全书打下基础。讨论各种类型的硬件以及它们对硬件/固件接口的影响。该章定义了原则和最佳实践、目标读者以及产品生命周期。此外，还展示了一个贯穿全书的案例研究。

第 2 章：该章介绍了七项原则并高度概括了本书的讲述方向，以及本书按照这个方向来讲述的原因。理解这些原则是理解列出的实践为什么是最佳实践的关键。

第 3 章：硬件工程师和固件工程师之间合适而充分的合作是影响嵌入式产品成功的关键因素。该章定义了在合作中的角色和过程。

第 4 章：在项目开始之前，必须做计划以确定并同意新产品采取的研究方向。该章涵盖了在计划新项目时所需考虑的几个方面。

第 5 章：大多数被指派去编写文档的工程师都不喜欢这项工作。同时，大多数工程师都因为阅读不完整且不正确的文档而迷糊。该章讨论文档的类型、何时编写文档、如何检查文档以及文档应包含哪些类型的内容。

第 6 章：该章介绍了可以完成在其自己所在领域内所有任务的组件的概念，并且讨论了超级组件的优点以及如何建立并根据需要使用超级组件。此外，还讨论了实际实现时的局限以及处理方法。

第 7 章：该章讨论设计的各个方面，例如事件、上电顺序、通信及控制。

第 8 章：寄存器是硬件与固件间的基本接口。该章详细讨论寄存器，包括地址、数据位的位置以及数据位的类型。

第 9 章：由于业界使用的中断设计缺乏一致性，因此该章着重于详细讨论如何管理硬件产生并传递至固件的中断。该章还将提议一种中断标准并对此展开详细讨论。

第 10 章：通常，很少考虑错误以及如何修复错误。该章讨论必要的设计元素以便固件能中止硬件的操作、恢复及继续处理。

第 11 章：逻辑分析仪无法探测芯片的内部，但是了解芯片内部发生的情况对于确保固件能在硬件上正常运行至关重要。在芯片内安放一些固件可访问的钩子能使固件获取信息供工程师分析。该章将介绍许多可以运用的钩子。

第 12 章：该章总结全书，还包含几个卡通图画来演示书中所讲述的概念。

**附录**

附录 A：该附录将本书中所有的最佳实践汇聚于此。

附录 B：该附录解释和说明第 5 章涉及的文档模板。

附录 C：对于必须共同完成某项目的硬件和固件工程的学生该附录就如何将本书中的知识传达给他们提供了一些建议。

附录 D：由于本书涉及两个不同的工程学科：硬件工程和固件工程，因此本附录涵盖了其中一个领域不为另一个领域所理解的术语。

# 本书约定

本书正文中的大部分内容是在讨论概念，散布其中的则是下列一个或多个元素：图、程序清单、寄存器映射、最佳实践和实战故事。

# 图

图 0-1 是一个例图。

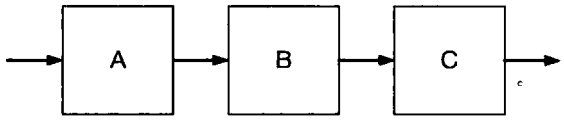



图 0-1　例图

# 固件程序清单

程序清单 0-1 是固件的 C 语言源代码的示例。

**程序清单 0-1　C 语言代码示例**

```
/* Read the current list of pending interrupts */
interrupts = *interruptRegister;
```

# 硬件电路

本书中列出了一些硬件电路，并给出了原理图及其等价的 Verilog 代码。图 0-2 是原理图，程序清单 0-2 是示例电路对应的 Verilog 代码。

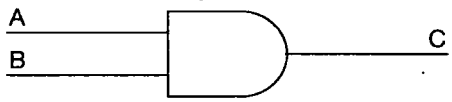



图 0-2　示例电路的原理图

**程序清单 0-2　示例电路的 Verilog 代码**

```
// A simple AND gate.
assign c = a & b;
```

# 寄存器映射

本书采用图解形式讨论寄存器、如何将各种数据位映射到寄存器中以及运行模式和这些数据位的复位值。有关该格式的详细解释参见第 5 章。

表 0-1　寄存器表格示例

| | MSB | | | | | | | LSB |
|---|---|---|---|---|---|---|---|---|
| | 常用寄存器—0x0004 | | | | | | | |
| 位 | 31 30 29 28 | 27 26 25 24 | 23 22 21 20 | 19 18 17 16 | 15 14 13 12 | 11 10 9 8 | 7 6 5 4 | 3 2 1 0 |
| R/W | - - - - | - - - - | - - - - | - - - - | - - - - | - - - - | - - - - | - C B A |
| 复位 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 |

A：本数据位的第一种含义。

B：本数据位的第二种含义。

C：本数据位的第三种含义。

# 最佳实践

本书包含了与所讲授概念相关的300多个最佳实践。除了在全书中展示这些最佳实践外，还将它们统一收集在附录A中，从而提供了一个简明的检查表，可在芯片项目的设计期间使用。此外，还可以在出版社网站(elsevierdirect.com/companions)及作者的网站(garystringham.com/hwfwbook)下载到以电子表格形式给出的这些最佳实践。

在本书正文、附录A及电子表格中的每个最佳实践都有一个形式为X.Y.Z的ID编号。

> ☑ **最佳实践**
>
> 1.1.1 *硬件/固件接口设计的最佳实践。*

与本书一样，Excel电子表格数据库也是有版权的材料。购买本书的读者有权(并且鼓励这么做)以这个数据库作为起点并将其修改成适合自己设计小组需要的形式，但是要遵守一些约定。有关该数据库及版权使用许可的细节内容请参见附录A。

> ☑ **最佳实践**
>
> 1.1.2 *Copyright © 2009，Gary Stringham & Associates LLC.保留所有权利。不要将此分发到您所在的设计小组以外。*

# 实战故事

散布在全书中的这些内容都是真实的故事，它们将有助于展示所在主题的影响。这些故事都来自于进行设计和解决问题的一线工程师(大部分是我自己)。下面就是这种故事的示例(并非真实发生的故事)。

> **实战故事**
>
> *我记得从一位朋友的朋友那里听过一个故事，他听一位工程师说有一位经理告诉她的下属……*

# 合作站点

从本书的合作站点 elsevierdirect.com/companions/9781856176057 上能找到包含了300多个最佳实践的电子表格数据库的链接，第5章讨论的文档模板，以及其他相关的内容。

请访问我的网站 garystringham.com/hwfwbook 以获得相同的工具和我在这个领域的工作有关的链接，以及如何与我直接联系的细节信息。

## 联系作者

如果您有任何有关本书或有关硬件/固件接口设计方面的问题，请随时通过 gary@garystringham.com 联系我。

## 致谢

我要感谢 Jack Meador 和 Mike Merrell，这两位硬件工程师在我从事项目工作时忍受了我不断提出的疑问、问题和请求，这个工程也促使我写成本书。他们在各自的硬件领域内提供了有价值的看法和帮助。他们及其公司以及其他公司的硬件工程师和固件工程师，提供了大量可用于很多最佳实践的材料，以及本书中的实战故事的材料。

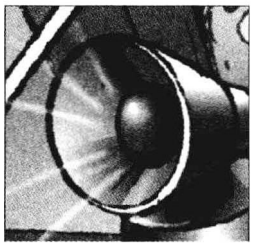
我还要感谢我的直接经理 Warren Johnson 和 Tracy Sauerwein 以及他们的上级经理 Sandy Lieske 和 Von Hansen。本书的最终出版离不开他们无私的支持，是他们伴随我从生疏到熟练的成长过程，使我受益良多。

在我的科技文写作教练 Joel Saks 的耐心指导下，我糟糕的写作水平提高了很多。他具有用只言片语就超越我所及的能力。此外，他还对我提出的概念进行了严格分析，并促使我进行清楚而准确的表达，从而使我在表达时能够证明对我自己而言是显而易见的那些概念是合理的。

我还要感谢 John Blyler、Clive “Max” Maxfiled、Jack Meador、Mike Merrell、Joel Saks 以及其他三位人士(他们希望保持匿名)审阅本书的全文或部分内容。他们的建议都极具价值，促使本书愈加完善。感谢 Mike Merrell 帮助完成 Verilog 代码，感谢 Kevin Falk 绘制了汽车插图。还要感谢 5 年来向我提出建议并给予热情鼓励的很多朋友，是他们的鼓励促使我完成本书的撰写。

最重要的，我要感谢我的妻子和孩子们  ，感谢他们的宽容和长期忍受我花费许多夜晚及周末的时间来撰写本书，却从未修理房子、带孩子参加活动或者和全家旅游。

# 目　录



# 第1章

# 引　言

每当硬件工程设计小组和固件工程设计小组试图合作时，他们总会遇到各种问题和冲突。其原因在于他们的开发环境不同、有不同的工具集，使用的术语也不同。他们通常在同一家公司的不同部门或者是不同的公司工作。他们要合作，但是常常在工作流程和方法上出现矛盾。由于设计好的硬件和固件必须成功集成后才能生产出最终的产品，因此我们必须合理地设计硬件/固件接口，包括人员、技术规范、工具和技术等诸多因素。

嵌入式系统的本质决定了硬件设计总是先于固件设计。尽管目前也涌现出一些工具和技术允许它们之间可以更多地并行操作，但是最终，硬件必须率先完成，之后固件设计小组才能进行最终的开发和测试工作。虽然目前业界已经投入大量努力以确保能正确地设计出硬件/固件接口，然而当硬件和固件进行系统集成时，总有问题存在。

与硬件相比，固件出现的问题相对比较容易解决。因为重新流片一块专用集成电路芯片(ASIC 芯片)[1]要耗时近 4 个月、花费几百万美元，而且根据芯片的制造工艺不同(如 90 纳米、65 纳米等)，其费用也有所不同。所以，如果硬件出现问题，固件设计小组就会“代为受过”承担压力，找到解决硬件问题的变通方法，避免拖延进度和增加成本。嵌入式系统专家 Jack Ganssle 曾幽默地说：“质量问题总是固件的错，因为修复硬件已经来不及了。”

硬件芯片比较昂贵，而且设计和投片的难度较高；出于商业需要，有必要完全弄懂芯片。为此，从方便固件工程师的角度来设计硬件芯片就成为系统能够正常运行的关键。本书作者基于多年来为硬件芯片编写固件的经验，提供了全面研究硬件芯片设计的方法。体现在本书中的这些研究成果包括许多实用且明智的想法，将结构化与严谨的设计思想应用于设计中。本书旨在为硬件工程师和固件工程师提供设计原则和最佳实践，从而提高其嵌入式系统的开发和集成能力。本书对于嵌入式产品开发阶段最为适用，特别适合于既要开发产品的芯片又要开发该产品的固件的情况。

第 1 章为全书的主题提供背景知识，并为本书其他章节提供论述基础。第 2 章讨论硬件/固件接口设计的七项原则。本书的其余部分包含了超过 300 个最佳实践。显然，本书列出的这些最佳实践并不能覆盖该领域的所有问题。不过，当您读完本书接下来的其他章节之后，也许会想起您使用过的最佳实践并从他人中获得灵感。请把它们记下来，然后应用到您的工作中去，并与其他人分享您的发现。

---

1. 在本书中，术语 ASIC 涵盖范围较宽，除了用于“专用集成电路芯片”之外，还包括“专用标准产品(ASSP)”、“片上系统(SoC)”、“现场可编程门阵列(FPGA)”等。详情见本章后面的定义。

# 1.1 硬件/固件接口

硬件/固件接口就是硬件与固件之间相互接触、相互通信的连接点。从硬件的角度来看，硬件/固件接口表现为一系列可被固件通过读和写访问的可寻址寄存器，同时还包括用于给固件发出事件通知的多个中断。从固件的角度看，硬件/固件接口表现为设备驱动程序或者某些底层软件，这些软件通过写入寄存器、解释从寄存器读出的信息，以及响应硬件中断请求等方法来控制硬件。当然，硬件不只是有寄存器和中断，而固件也不只是有设备驱动程序，但恰恰正是这些东西(寄存器、中断、驱动程序等)组成了硬件/固件接口，它们是工程师顺利地实现硬件/固件集成必须关注的东西。

在业界，术语“硬件”和“固件”的外延和内涵是不同的，所以，有必要在本书中先明确它们的定义。

## 1.1.1 硬件、芯片及组件

在电子工程领域中，术语“硬件”涵盖嵌入式产品中所有的电子电路，包括印制电路板、像电阻和电容这样的无源器件，以及芯片等。当然，它也可以用来表征各种非电气和机械部件，比如插销、垫片、防护外壳/外罩等。与此同时，术语“芯片”指的是使用硅材料或者其他半导体材料制造出来的任何物理设备，它主要由许多晶体管组成，而这些晶体管组合起来就实现了各种数字或模拟功能。它可以是简单的单一功能设备，也可以是复杂的多功能设备，这些多功能设备可能包含处理器、存储器、硬件接口电路或其他功能电路。

就本书而言，“硬件”和“芯片”专指器件(比如ASIC和FPGA)的某个子集，尤其是那些通过寄存器和中断与固件打交道的硬件器件。这些器件不包括微处理器、微控制器及存储器等。更进一步说，本书中的“硬件”和“芯片”是可互换的，比如“硬件小组设计芯片”。

“芯片”通常包含一个或多个功能“组件”，比如USB通信组件、MPEG压缩器、存储器控制器等。也有可能是某个特定组件的多个实例(副本)，比如两个UART组件。芯片内的组件通常要相互通信，并通过公共总线与外部存储器通信。每个组件往往都设计成一个单独的单元。当我们设计新芯片时，其中很可能包括以往设计中所用过的组件和新加入的组件。

在本书内容中，“芯片”是指多个系列的通用集成电路，虽然每个系列之间都存在各自细微的差别，不过“芯片”这个概念在绝大多数情况下都适用。

### 1. ASIC

ASIC(Application Specific Integrated Circuit，专用集成电路)用于某个特定品牌的特定产品中。它是多个标准组件和专有组件的定制化集成。ASIC 是在功耗、性能和成本等方面经过优化且批量生产的芯片。这意味着很可能同时使用多款不同的 ASIC，而每款 ASIC 都有各自的硬件设计小组和固件设计小组。这些硬件小组和固件小组必须持续协作，才能为不同的产品系列以及多代产品生产出各种不同的 ASIC。

硬件小组和固件小组有可能合作共同为某家公司生产产品，也有可能其中一个小组或

两个小组受雇为另外一家公司。无论哪种情况，设计小组之间都必须密切合作，以便给硬件小组提供更多的修改和改进设计的机会，因为硬件小组可以提前与固件小组合作。

### 2. ASSP

ASSP(Application Specific Standard Product，专用标准产品)与 ASIC 类似，唯一不同之处在于：ASSP 用于某个特定的应用领域，而且它的销售目标是客户群而不是某个特定客户——这也是命名中“标准产品”部分的由来。与此形成鲜明对比的是，ASIC 总是为某个特定客户设计和制造的。ASSP 只提供标准功能，这使得许多公司生产的多种嵌入式产品都可以直接使用 ASSP。这意味着由某家公司生产出某种 ASSP，同时会有多家公司为这种 ASSP 生产固件。不同厂家的各种版本的操作系统都需要设备驱动程序；当然，也可以针对既定的嵌入式系统中的固件开发设备驱动程序。因此，通常要牵涉来自多家公司的固件小组。通常情况下，固件小组都不会事先与设计 ASSP 的硬件小组直接接触。与生产 ASIC 的情况差不多，硬件小组与固件小组之间往往没有一对一的关系。

这种情况给硬件小组带来额外负担，因为他们总是试图满足许多不同固件小组的需求，但无法与每个固件小组都合作。在这种情况下，硬件小组应该在设计阶段就开始挑选一到两个固件小组来“搭伙”共同开发芯片。这样做的最终结果往往是：公司在出售芯片时可以搭载多款设备驱动程序发售，这也是最为常见的情形。这样，其他公司就可以立刻将其用于他们的产品中。

某些 ASSP 所提供的功能有时非常通用，以至于同一种设备驱动程序可以运行在多家公司出产的任何品牌的硬件产品上。比如兼容 EHCI(Enhanced Host Controller Interface，增强型主机控制接口)的 USB(Universal Serial Bus，通用串行总线)控制器、兼容 16650 协议的 UART(Universal Asynchronous Receiver Transmitter，通用异步收发器)等就是这样。这意味着寄存器、数据位与功能都非常一致。这也使得同一个设备驱动程序可以运行在各种不同品牌的 ASSP 上。要保持这种兼容性，就需要对硬件设计小组加以约束才行，这些约束甚至包括不允许他们擅自改进其他品牌产品的设计。

### 3. FPGA

FPGA(Field Programmable Gate Array，现场可编程门阵列)能够通过重新设计程序来改变器件的功能，从而具有很大的灵活性。但是，它们每个部分的功耗更大，性能更低，成本更高。

FPGA 可以烧录用户定制的混合内容。这种定制的混合使得它与 ASIC 类似，因为通常需要固件小组和硬件小组一对一地结对开发。然而，由于 FPGA 可以在几小时内修改，因此 FPGA 编程会存在很多版本。这需要硬件和固件小组的密切合作，才能确保固件的版本与 FPGA 程序的版本配对并能共同工作。

当产品部署给用户后还需要修改设计时，也会用到 FPGA。FPGA 支持发布新的编程文件并将其下载到产品中。

FPGA 还可以烧录标准的混合内容。这与 ASSP 类似，因为公司可以向很多客户销售相同的封装。这意味着会有很多固件小组为这个标准的基于 FPGA 的内容编写设备驱动程序。公司采用这种方法就能销售出少量的设计，且不会发生与将设计发送给工厂制造相关

的 NRE(NonRecurring Engineering，非重现工程)花销(正如那些不基于 FPGA 的 ASSP)。

本书中所提及的对重新流片芯片所需的时间和花销并不适用于 FPGA。固件无法区分 ASIC/ASSP 或者 FPGA，因为它们的寄存器/中断接口都是一样的。

### 4. SoC

SoC(System On Chip，片上系统)与 ASIC 及 ASSP 类似，但它除了包括组件的混合体，还包含有一个或多个处理器，还可能有存储器。它既可以是用户定制的混合体(ASIC+处理器)，也可以是标准的混合体(ASSP+处理器)。可以将它发送至工厂进行制造，或者烧录在 FPGA 上。

对于本书，SoC 和 ASIC、ASSP 以及 FPGA 都是同义词。本书不讨论 SoC 的处理器和存储器部分；取而代之，讨论的焦点是芯片的其余部分——固件可访问的功能组件。尽管固件要在处理器上执行、驻留在存储器上并且访问存储器，但是本书并不涉及如何设计处理器或存储器。

表 1-1 总结了各种芯片的不同点，它由三个部分组成。

**表 1-1　各种芯片的不同点的总结**

| 第 1 部分 | ASIC | ASSP |
|---|---|---|
| 内容混合 | 用户定制的 | 标准的 |
| 目标客户 | 一个 | 多个 |
| 固件小组 | 一个 | 多个 |
| 设计优化 | 针对某一个 | 针对所有 |
| 设计变化/改进 | 可以 | 很难 |
| **第 2 部分** | **制造的部件** | **可编程部件(FPGA)** |
| 可编程能力/灵活性 | 无 | 有 |
| 重新流片的成本 | 高 | 无 |
| 相对功耗 | 低 | 高 |
| 相对速度 | 快 | 慢 |
| 相对单件成本 | 低 | 高 |
| 相对最初的 NRE 成本 | 高 | 无 |
| 销售规模 | 高 | 低 |
| **第 3 部分** | **非 SoC** | **SoC** |
| 微处理器 | 无 | 一个或多个 |

### 5. 其他芯片

还有其他类型的芯片，比如 DSP(Digital Signal Processor，数字信号处理器)和 CPLD (Complex Programmable Logic Device，复杂可编程逻辑器件)，但是它们的基本原理是相同的。固件要与这些器件进行交互，控制并响应这些器件。